IN THE SPECIFICATION

Paragraph [0038] is sought to be rewritten as follows:

Any parameter we are interpolating across a triangle (or other primitive) is guaranteed to be in the range 0 . . . 255 for interior points to the triangle. Exterior points can (and will) have parameter values outside this range. The farthest point outside of the primitive we will ever calculate a value for is 7x1.414 [[pixelsl]] pixels so allowing an extra three bits on the range of the calculations allows for this.

Paragraph [0040] is sought to be rewritten as follows:

where xt and yt are the coordinates of the tile in question, xto and yto are the coordinates of the tile where the c parameter (called pStart) is calculated. The xto and [[ytovalues]] yto values are taken from the first tile's x and y [[coordiantes]] coordinates after any new parameters have been received.

Paragraph [0046] is sought to be rewritten as follows:

The x*dpdx calculation (or y*dpdy) calculation can be done using 4 multipliers [[1]] but as the multiplicands are linearly related with a successive difference of one the following equations may provide a more economical implementation:

0*dpdx = 0
1*dpdx = dpdx
2*dpdx = (dpdx<<1)
3*dpdx = (dpdx<<1)+dpdx

Paragraph [0055] is sought to be rewritten as follows:

Existing rasterizers are fixed function devices. With the advent of multi texturing it has become impossible to cast sufficiently flexibility into a fixed function device, particularly when up to 8 textures can be combined in one fragment. Microsoft have recognized this in DX8 and are pushing programmable shading languages as the way forward. Clearly the 3D chip community [[have]] <u>has</u> no choice [[by]] <u>but</u> to go along with this.

Paragraph [0074] is sought to be rewritten as follows:

The more data the memory controller can read or write per request the more efficient it will be able to run. Needless to say you should strive to make use of all the data in the transfer and not some small fraction of it. Tiles are also visited in an order aimed at promoting optimum memory usage, although the Memory Controller can [[hid]] <u>hide</u> the page break cost in all transfers larger than one (byte wide) tile. More extensive caching techniques are used to smooth out demand peaks and to allow some degree of pre-fetching to occur.

Paragraph [0084] is sought to be rewritten as follows:

As any context switchable state flows through into the rasterizer part it goes through [[is]] the Context Unit. This unit caches all context data and maintains a copy in the local memory. A small cache is needed so that frequently updating values such as mode registers do not cause a significant amount of memory traffic. When a context switch is needed the cache is flushed and the new context record read from memory and converted into a message stream to update downstream units. The message tags will be allocated to allow simple decode and mapping into the context record for both narrow and wide messages. Some special cases on capturing the context as well as restoring it will be needed to look after the cases where multiple words are mapped to the same tag, for example as used when program loading. One of the side effects of this is to be able to remove the context logic in each unit and the readback mechanisms (you could just read directly from context record in memory). Also the previous context mechanisms are problematic in the texture pipes (because the message stream doesn't run through the pipes) and this solution handles this transparently. This will be very fast as changing context will only require a small amount of state to be save (from the cache) and the restore will be at 1 message per cycle (even for wide messages). By allowing wide message loading of the LUTs, WCS, etc. the context restore could probably be reduced to 500 cycles or 3 microseconds.

Paragraph [0111] is sought to be rewritten as follows:

Output DMA: The output DMA is mainly used to load data from the core into host memory. Typical uses of this [[is]] are for image upload and returning current vertex state. The output DMA is initiated via messages which pass through the core and arrive via the Host Out Unit. This allows any number of output DMA requests to be queued.

Paragraph [0114] is sought to be rewritten as follows:

The Current Parameter Unit's main task it to allow a parameter such as a color or a texture to be supplied for every vertex even when it is not included in a DMA buffer. This allows vertices in OpenGL to inherit previously defined parameters without being forced to supply them on every vertex. Vertex arrays and vertex buffers always supply the same set of predefined parameters per vertex. Always supplying 16 sets of parameters on every vertex will [[reducing]] reduce performance considerably so the Current Parameter Unit tracks how many times a parameter is forwarded on and stops appending any missing parameters to a vertex once it knows the Vertex Shading Unit has copies in all its input buffers.

Paragraph [0118] is sought to be rewritten as follows:

The coordinate results are passed to the Vertex Machine Unit via the message stream and the 16 parameter results are passed directly to the Geometry Unit on a private bus. The two output ports allow for a higher vertex throughput.

Paragraph [0120] is sought to be rewritten as follows:

The Cull Unit caches the window coordinates for the 16 vertices and when a Geom*message arrives the unit will use the cached window coordinates to test clip against the viewing frustrum and, for triangles, do a back face test. Any primitives failing these tests (if enabled) will be discarded. Any primitives passing these tests are passed on, however if the clip test is inconclusive the primitive is further tested against the guard band limits. A pass against these new limits means that it will be left to the rasterizer to clip the primitive while it is being filled--it can do this very efficiency and spends very little time in `out of view` regions. A fail against the guard band limits or the near, far or user clip plane will cause the primitive to be geometrically clipped in the Geometry Unit.

Paragraph [0133] is sought to be rewritten as follows:

The Parameter Set Up Unit is replicated in each texture pipe so it only does the set up for primitives which reach that pipe. The parameters handled by this unit are 8 four component color values and 8 four component texture values. For small primitives the performance of the 4 Parameter Set Up Units will balance the single Depth Set Up Unit. The vertex store in this unit is arranged as a circular buffer which can hold 48 parameters. This is considerably smaller than the 256 parameter store required to hold 16 parameters for 16 vertices. In most cases there will only be a few parameters per vertex so we get the benefit of being able to hold 16 vertices, but as the number of parameters per vertex increased then the total number of vertices which can be held will reduce. In the limit we can still hold all 16 parameters for three vertices which is the minimum number of vertices necessary to set up the plane equations. Color parameters can be marked as being `flat` when flat shading is enabled.

Paragraph [0135] is sought to be rewritten as follows:

All parameter calculations are done by evaluating the plane equation directly rather than using DDAs. This allows the tiles all primitives are decomposed into to be visited in any order and evaluation for fragment positions within a tile to be done in parallel (when needed). The origin of the plane equation is relocated from (0, 0) to the upper left fragment of a tile which overlaps the primitive [[so]] to constrain the dynamic range of the c value in the plane equation.

Paragraph [0166] is sought to be rewritten as follows:

Antialiased points are processed in a different way as it is not possible to use the edge function generators without making them very expensive or converting the point to [[an]] <u>a</u> polygon. The method used [[it]] <u>is</u> to calculate the distance from each subpixel sample point in the point's bounding box to the point's center and compare this to the point's radius. Subpixel sample points with a distance greater than the radius do not contribute to a pixel's coverage. The cost of this is kept low by only allowing small radius points hence the distance calculation is a small multiply and by taking a cycle per subpixel sample per pixel within the bounding box. This will limit the performance on this primitive, however this is not a performance critical operation but does need to be supported as the software has no way to substitute alternative rendering commands due to polymode behavior.

Paragraph [0176] is sought to be rewritten as follows:

Each texture pipe works autonomously and will compute the filtered texture values for the valid fragments in the tile it has been given. It will do this for up to eight sets of textures and pass the results to the Shader Unit in the pipe, and potentially back to the Texture Coordinate Unit for bump mapping. Processing within the texture pipe is done as a mixture of SIMD units (Texture Coordinate Unit and Shading Unit) or one fragment at a time (Primary Texture Cache Unit and Texture Filter Unit) depending on how hard <u>it is</u> to parallelize the required operations.

Paragraph [0179] is sought to be rewritten as follows:

The Texture Address Unit calculates the address in memory where the texel data resides. This operation is shared by all texture pipes (to [[saves]] <u>save</u> gates by not duplicating it), and in any case it only needs to calculate addresses as fast as the memory/secondary cache can service them. The texture map to read is identified by a 3 bit texture ID, its coordinate (i, j, k), a map level and a cube face. This together with local registers [[allow]] <u>allows</u> a memory address to be calculated. This unit only works in logical addresses and the translation to physical addresses and handling any page faulting is done in the Memory Controller. The layout of texture data in cube maps and mip map chains is now fully specified algorithmically so just the base address needs to be provided. The maximum texture map size is 8Kx8K and they do not have to be square or a power of two in size.

Paragraph [0189] is sought to be rewritten as follows:

The primary cache is divided into two banks and each bank has 16 cache lines, each holding 16 texels in a 4x4 patch. The search is fully associative and 8 queries per cycle (4 in each bank) can be made. The replacement policy is LRU, but only on the set of cache lines not referenced by the current fragment or fragments in the latency FIFO. The banks are assigned so even mip map levels or 3D slices are in one bank while odd ones are in the other. The search key is based on the texel's index and texture ID not address in memory (saves having to compute 8 addresses). The cache coherency is only intended to work within a sub tile or maybe a tile and never between tiles.[[2]]